Systems and Internet Infrastructure Security

Network and Security Research Center
Department of Computer Science and Engineering
Pennsylvania State University, University Park PA

Securing End-to-End Provenance: A Systems and Storage Perspective

Kevin Butler, University of Oregon
Patrick McDaniel, Stephen McLaughlin, and Devin Pohly, Pennsylvania State University
Radu Sion and Erez Zadok, Stony Brook University
Marianne Winslett, University of Illinois

HEC FSIO 2010 Workshop, Washington DC

Provenance

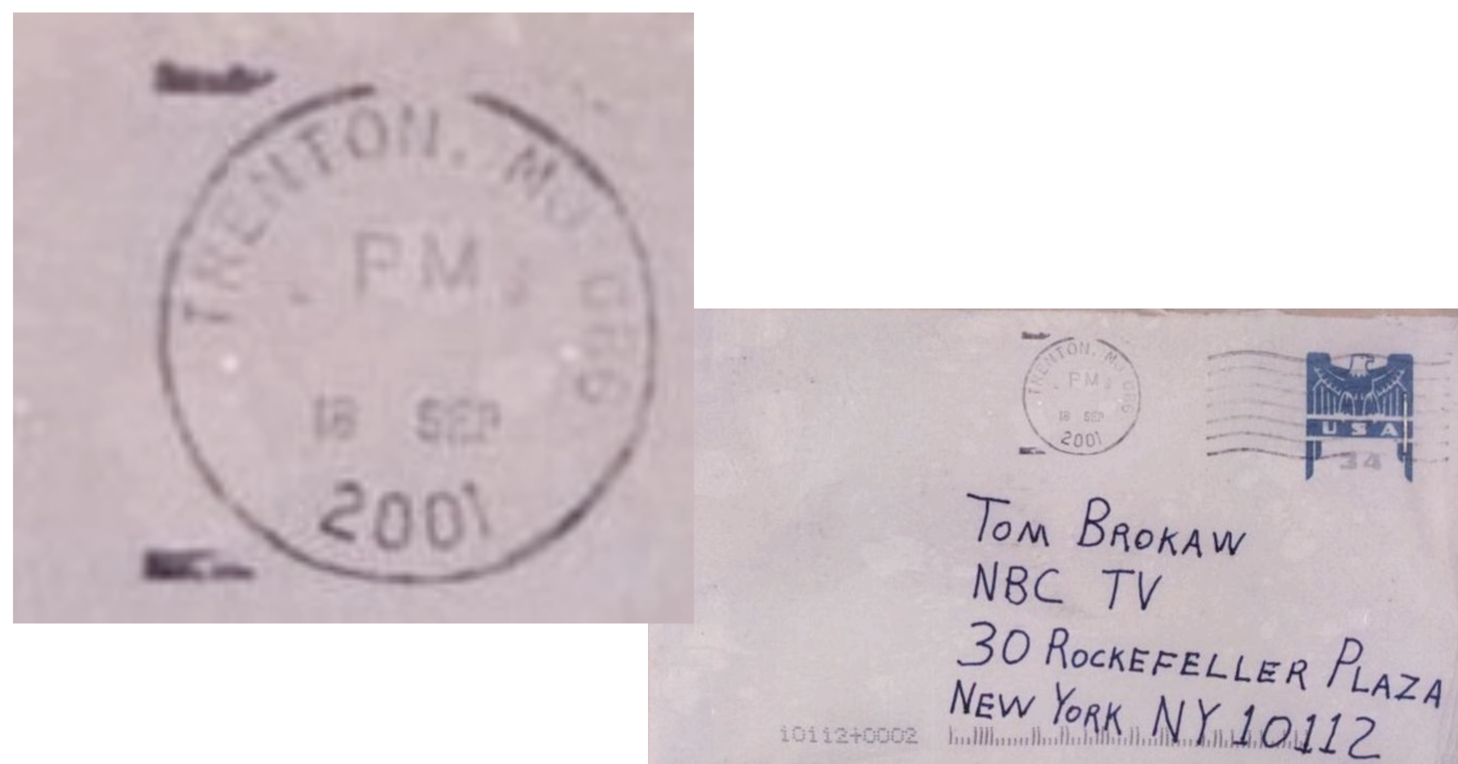
- Shuttle launch relies on thousands of systems and millions of parts all working together correctly: enormously complex
- File systems for HEC are similarly complex - originates from *many sources* and synthesized by *complex and sometimes hidden* processes
- What does the data mean and how can we interpret and analyze it?

Data Provenance

- Data provenance allows us to answer the following questions about the origin of data:
  - *Who or what* contributed to the generation of this data?
  - *What* is the data based upon?
  - *When* was the data generated?
  - *Why* was it generated?
  - *How* was it generated?
- A history of the object from origin through subsequent modifications is evidenced by a *provenance chain* (DAG)

End to End Provenance System

- Why another provenance collection system?
  - Strong security guarantees
  - Distributed provenance collection
  - EEPS will achieve the above two goals efficiently in high end computing systems



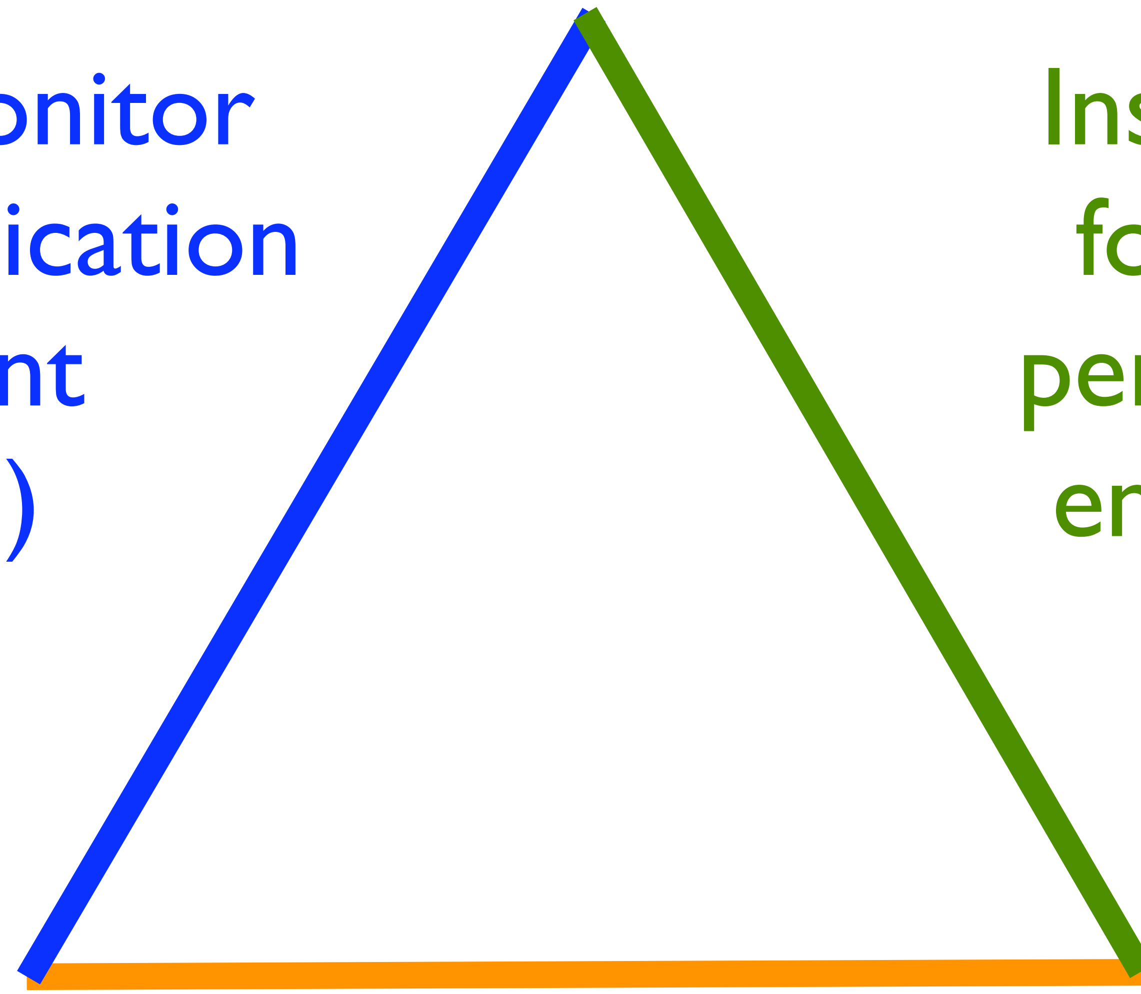
Secure Provenance Collection

- Provenance monitor (PM) analogous to reference monitor concept
- Three guarantees
  - Complete mediation
  - Tamperproofness
  - Verifiability
- Beyond authentication of records
  - Integrity/Trustworthiness of recording instrument and provenance-enhanced applications

Project Initiatives

Provenance monitor system and application development (McDaniel)

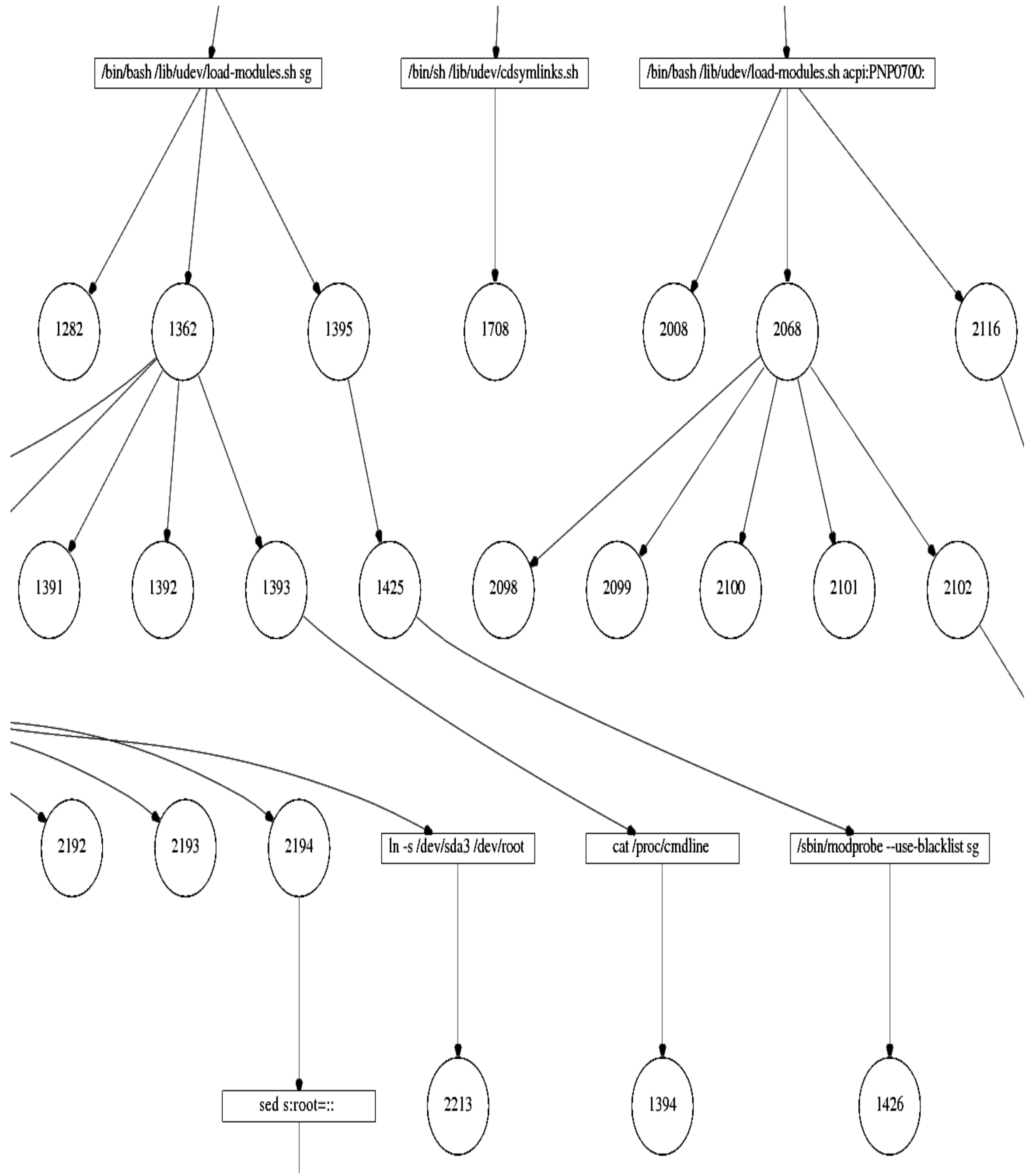
Instrumentation for measuring performance and energy (Zadok)



Provenance chain constructions and query management (Sion, Winslett)

Provenance Monitor

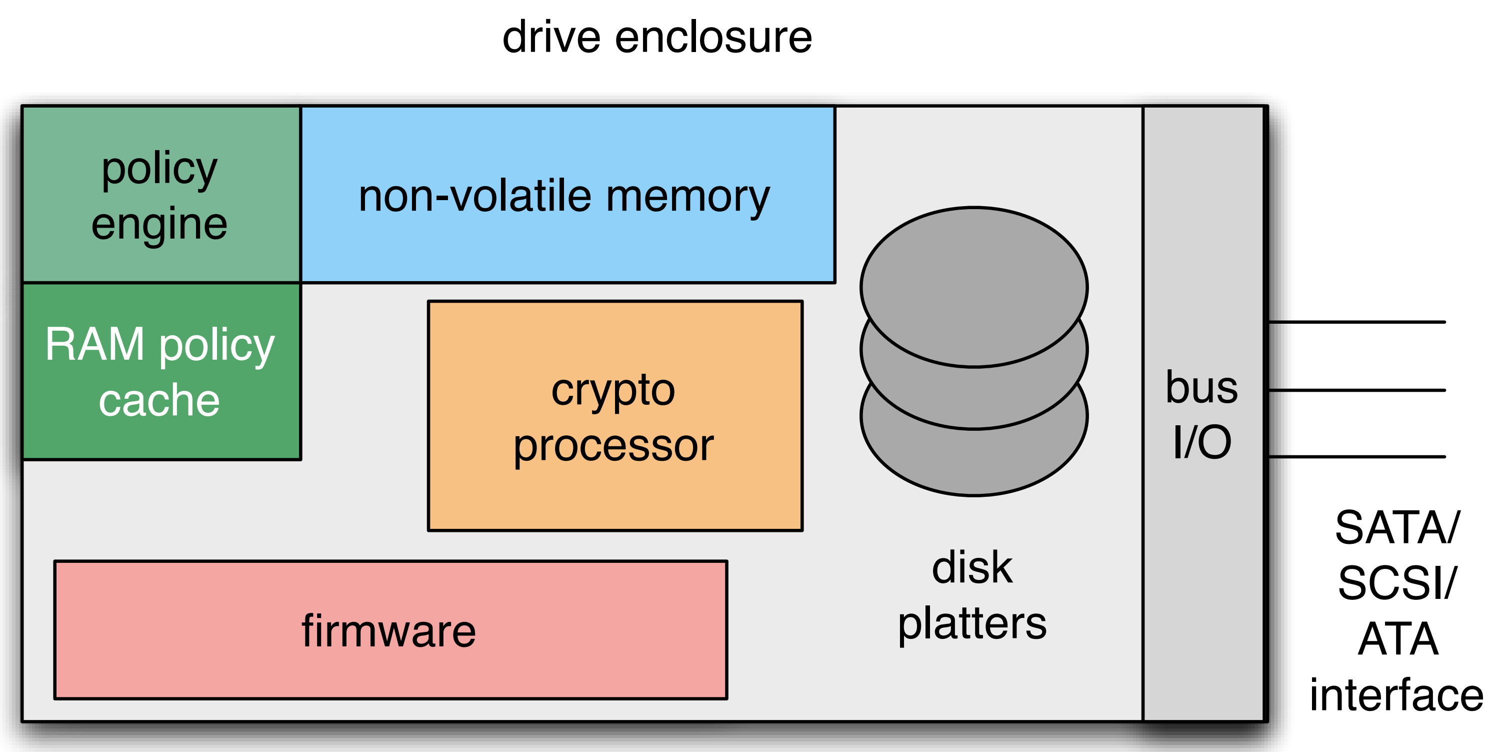
- Implementing LSM-based provenance monitor
  - LSM for complete mediation, tamperproofing
- Tracking provenance of entire VM runs
  - Created graph of entire process ancestry
  - Investigated visualizations which included file reads/writes
- Exploring potential for secure multi-host and interdomain provenance



Autonomously Secure Disks

PENNSTATE 1855

- Enforce security perimeter at external I/O interface
  - How? Store all security metadata *including provenance information* within the drive itself



"New Security Architectures Based on Emerging Disk Functionality", IEEE Security and Privacy, Sept/Oct 2010

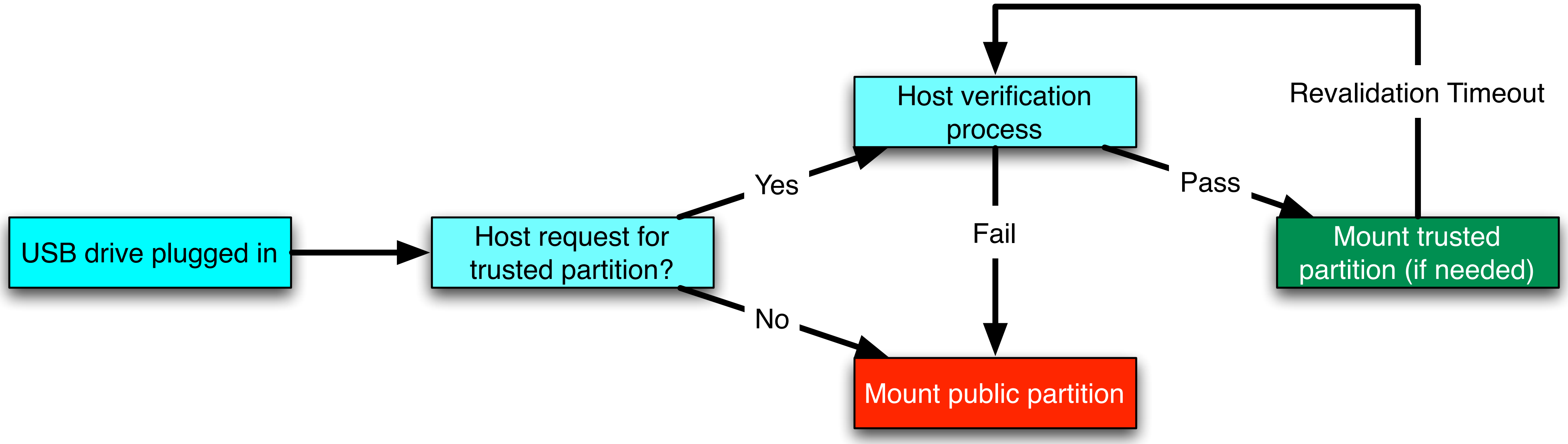
Host Validation

- Portable storage holds more and is more ubiquitous than every before (256 GB flash drives)
  - Public/private data on devices, want to share some info but protect other data

- How can we solve these issues?
  - Only allow registered devices on system
  - Virus scanning on flash drive
  - *General problem*: How do we know if the system we share data with is in a good (valid) state? Can we collect a provenance record relating to this data?

Kells

- USB storage device performs *attestations* with host in order to determine its integrity state
- Periodically repeat attestation to get continuous guarantee of host integrity
- Allow access to *trusted* partition only if system is in a good state



Continuous Attestation

- Support framework for runtime monitoring on system
  - Patagonix, Pioneer, BIND, LKIM, etc.
- Continuous attestation gives assurance that the system is in a good state
  - Length of time between attestations can be parameterized by an attestation period Δ_t
  - Acts as *security heartbeat*
- *Quarantine buffer* on storage device holds writes until system state is attested

Kells Security Properties

- **SEC:** *"Any read request completed by Kells was made while the host was in a good state."*
  - attestation received within Δ_t of the request and the system was not rebooted
- **INT:** *"Any write request completed by Kells was made while the host was in a good state."*
  - same dependency on attestation received within Δ_t as with **SEC**

$$
\begin{aligned}
(\mathbf{SEC}) \vdash \;\; \forall \;\; & (\mathtt{t}_{\mathtt{req}}, (l, n)), (\mathtt{t}_{\mathtt{att}}, sig), t \;\text{ s.t.} \\
& (\mathtt{t}_{\mathtt{req}}, (l, n)) = \mathsf{Recv}(\mathcal{D}) \;@\; t \\
\wedge \;\; & (\mathtt{t}_{\mathtt{att}}, sig) = \mathsf{Recv}(\mathcal{D}) \\
\wedge \;\; & e = \mathsf{SRead}(\mathcal{D}, (t, \mathtt{t}_{\mathtt{req}}, (l, n)), (\mathtt{t}_{\mathtt{att}}, sig)) \\
\supset \;\; & \mathsf{GoodState}(\mathcal{H}, (t, \mathtt{t}_{\mathtt{req}}, (l, n)), (\mathtt{t}_{\mathtt{att}}, sig))
\end{aligned}
$$

$$
\begin{aligned}
(\mathbf{INT}) \vdash \;\; \forall \;\; & (t, \mathtt{t}_{\mathtt{req}}, (l, n)), (\mathtt{t}_{\mathtt{att}}, sig) \;\text{ s.t.} \\
& (t, \mathtt{t}_{\mathtt{req}}, (l, n)) = \mathsf{Peek}(\mathcal{D}) \\
\wedge \;\; & (\mathtt{t}_{\mathtt{att}}, sig) = \mathsf{Recv}(\mathcal{D}) \\
\wedge \;\; & \mathsf{SWrite}(\mathcal{D}, (t, \mathtt{t}_{\mathtt{req}}, (l, n)), (\mathtt{t}_{\mathtt{att}}, sig)) \\
\supset \;\; & \mathsf{GoodState}(\mathcal{H}, (t, \mathtt{t}_{\mathtt{req}}, (l, n)), (\mathtt{t}_{\mathtt{att}}, sig))
\end{aligned}
$$

Bottom Line: Provides formal proof that data is protected.

Future Project Goals

- Revealing file access patterns
  - What is "least privilege?"
  - Forensic details of file and block access: "Which host accessed this particular data and where may that information have been disseminated?"
- Information flow systems
  - Provenance as enriched information flows
- Provenance calculus
  - Formalism for expressing and querying provenance data
  - Working toward more rigorous definitions of provenance
  - Potential for machine learning applications

Performance Enhancements

- Provenance monitor profiling
  - Enhanced profiling tools
  - Profiling provenance collection for workloads from scientific domains
  - EEPS calibration for a particular environment
  - LSM instrumentation
- Cost models for provenance collection
  - Hardware and storage requirements ($/GB)
  - New cost models based on types of provenance data collected and system architectures

UNIVERSITY OF OREGON

Securing End-to-End Provenance

Kevin Butler <butler@cs.uoregon.edu>

Systems Problems

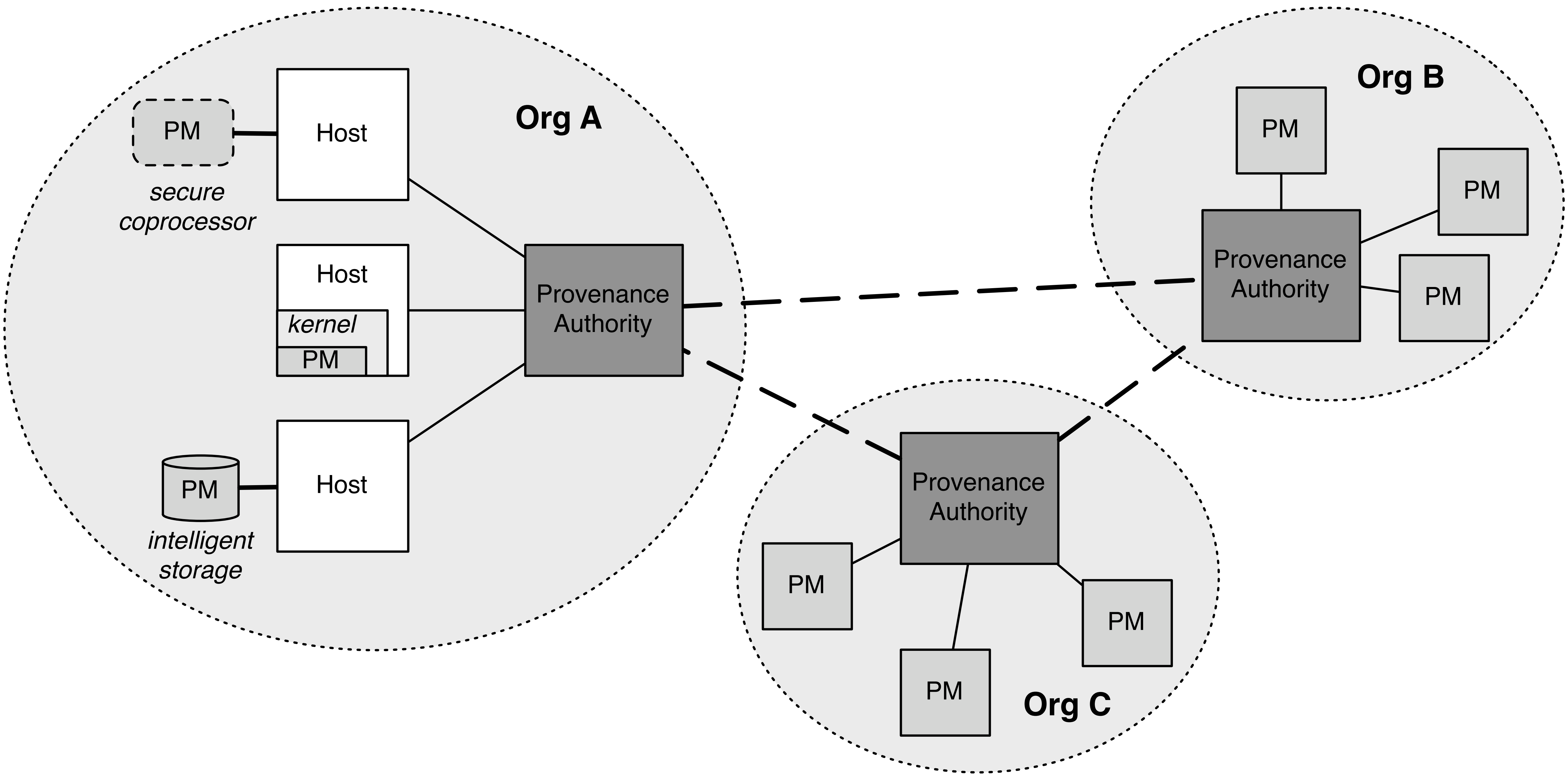
- Reliable high-volume data transmission from kernel to userspace
  - Currently using Linux relay mechanism
  - Investigating other means to increase reliability
- Inclusion of filenames in inode tracking
  - LSM provides little context here
  - Would provide additional information during analysis

Distributed PM

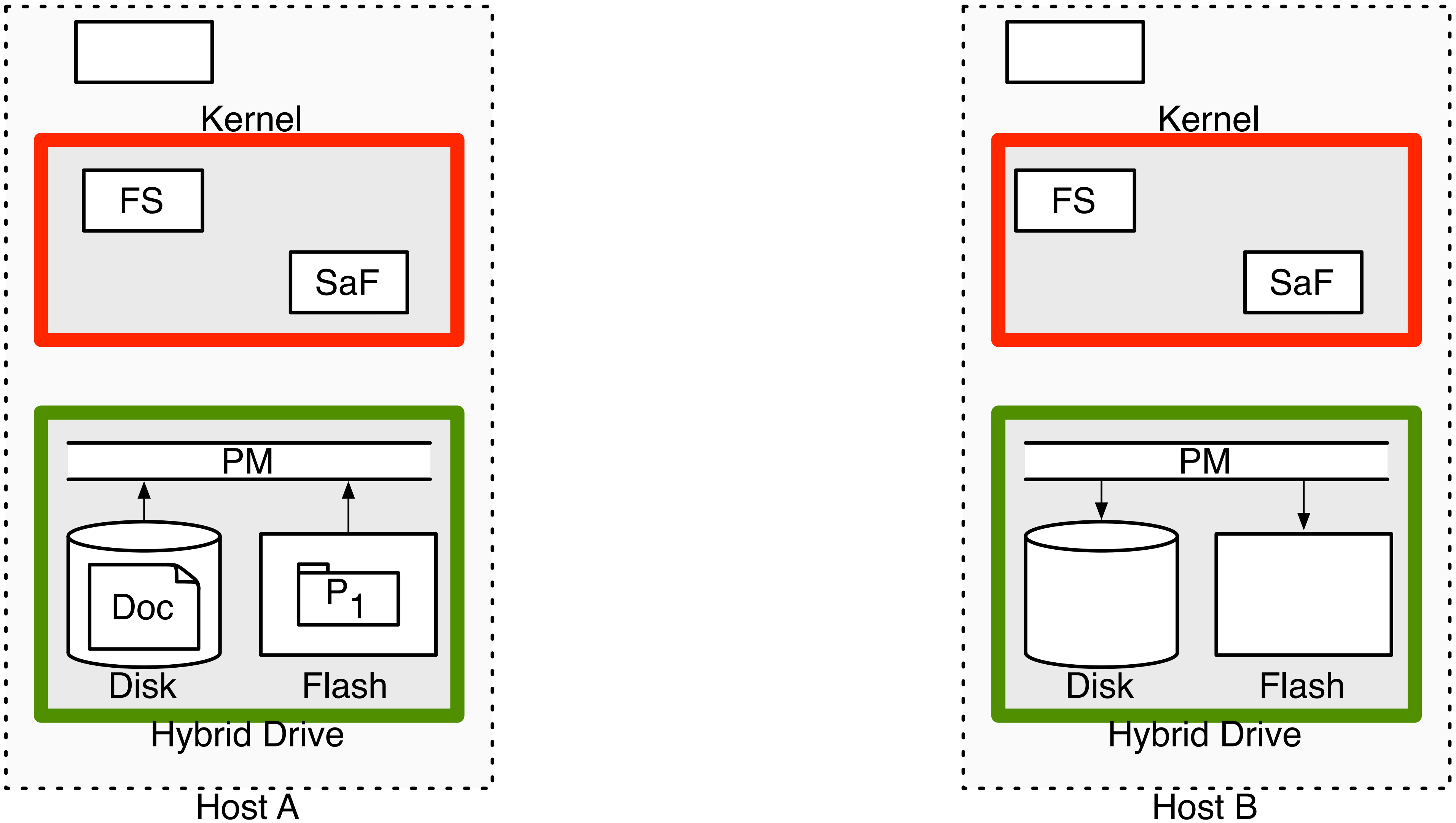
- Challenges in distributed provenance
- Domain specific policies for:
  - Auditors - confidentiality considerations
    - Cryptographic commitments [Hasan'09]
  - Divergent modification histories
    - Plausible version history
    - If necessary, plausible history may be checked against previous subjects in the ownership chain

Distributed Environments

PENNSTATE

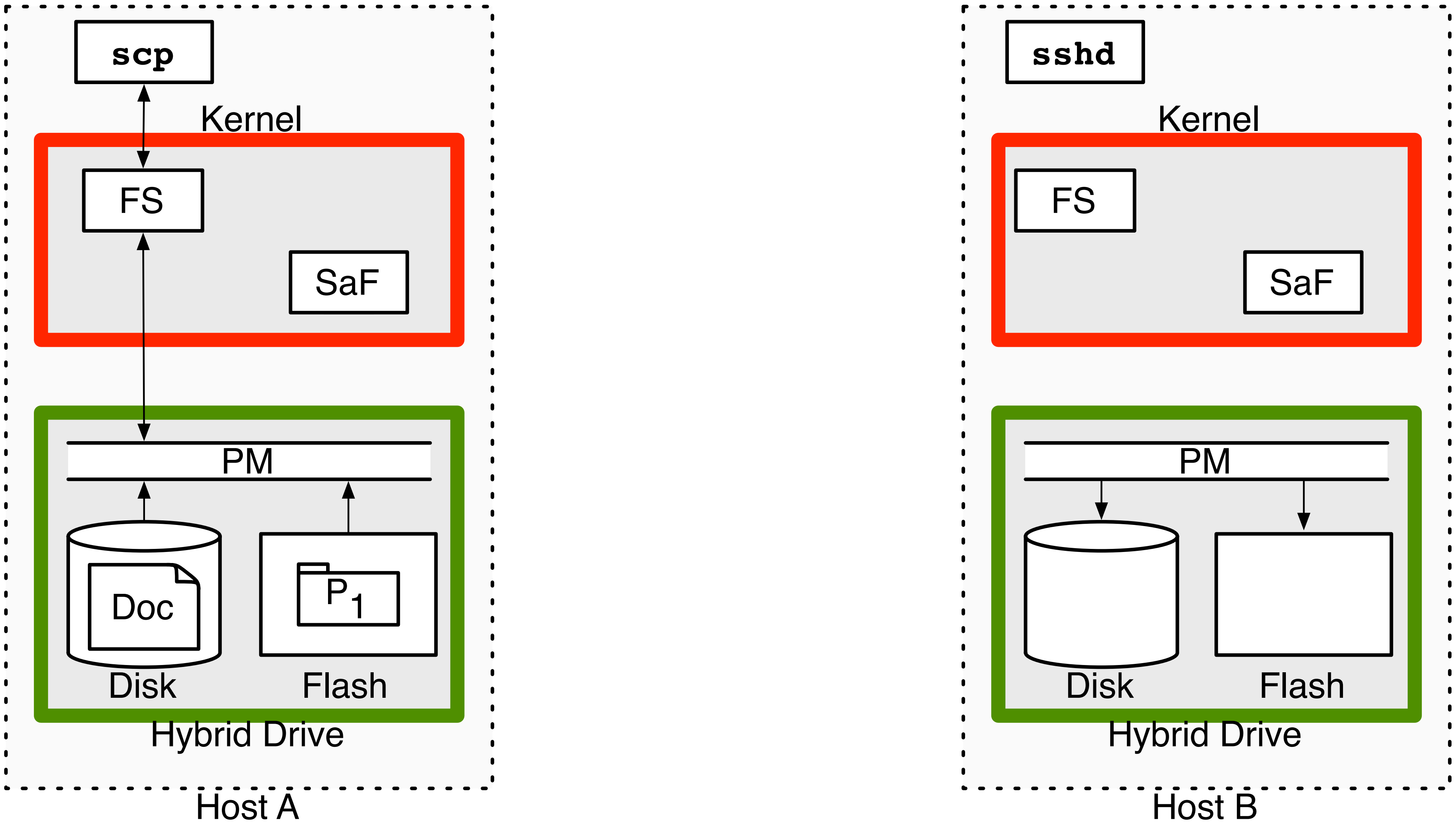


Distributed Example



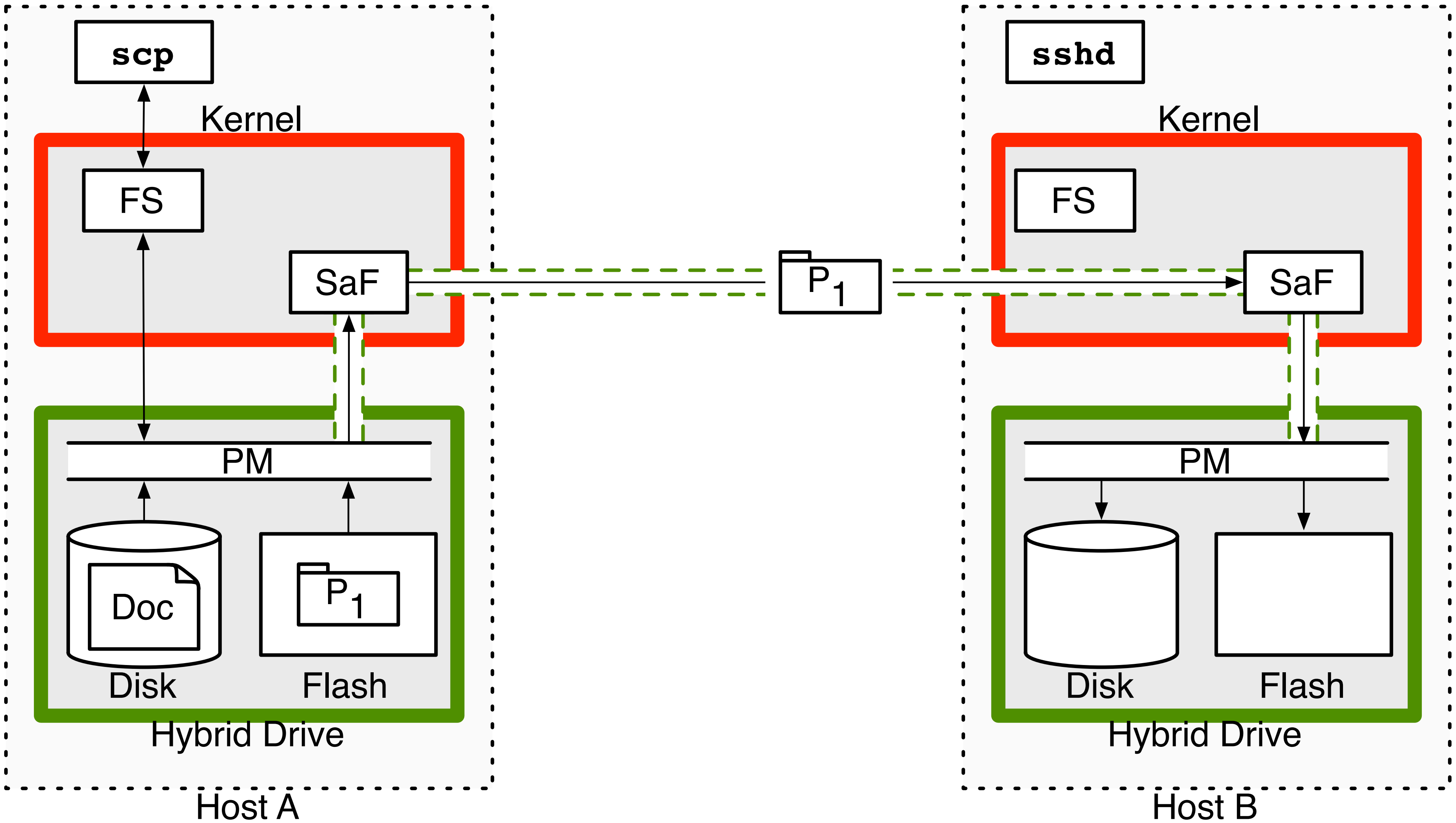
Example: File transfer between hosts with untrusted OSes and trusted storage

Distributed Example



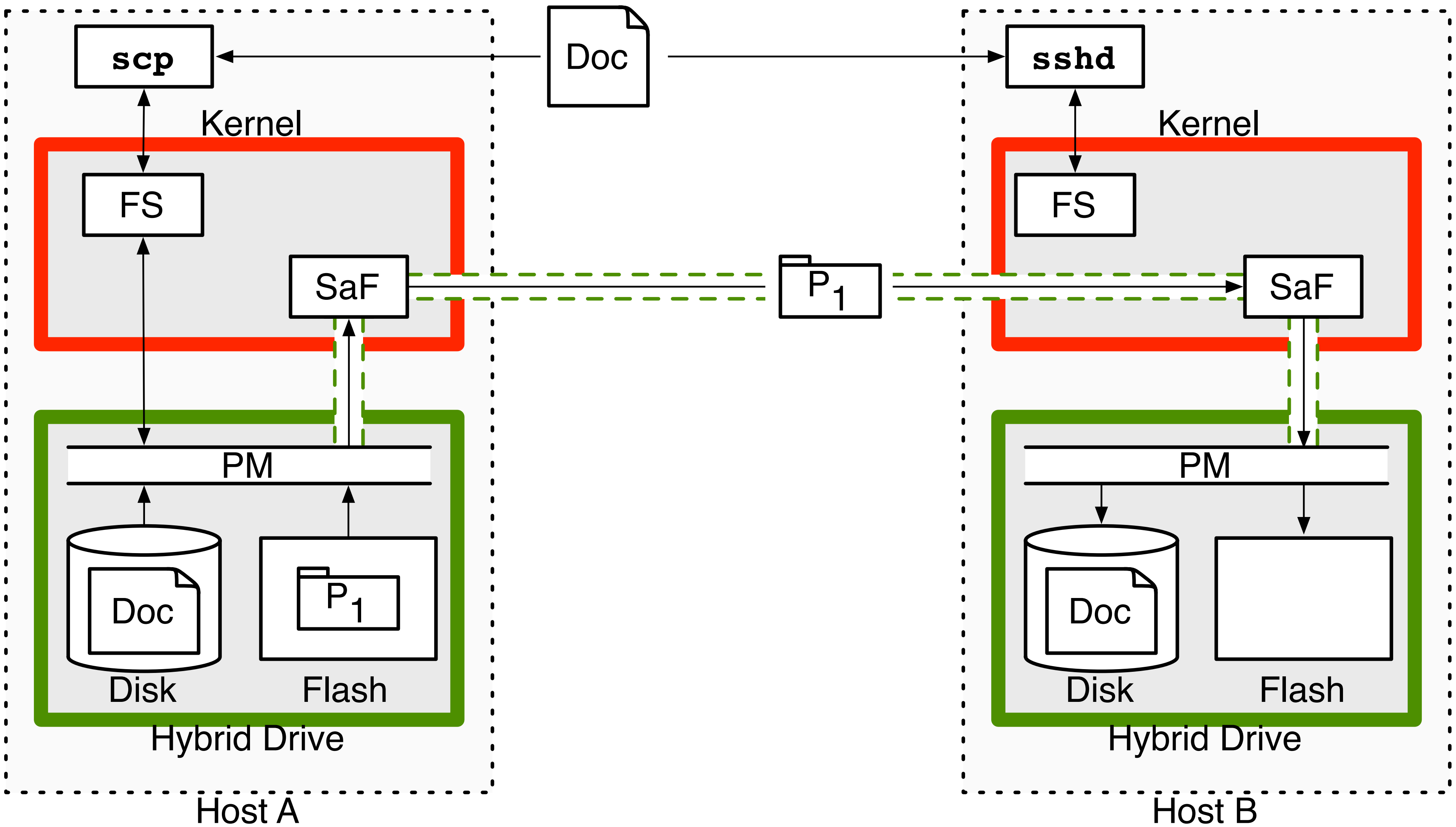
A program initiates a request for the file.

Distributed Example



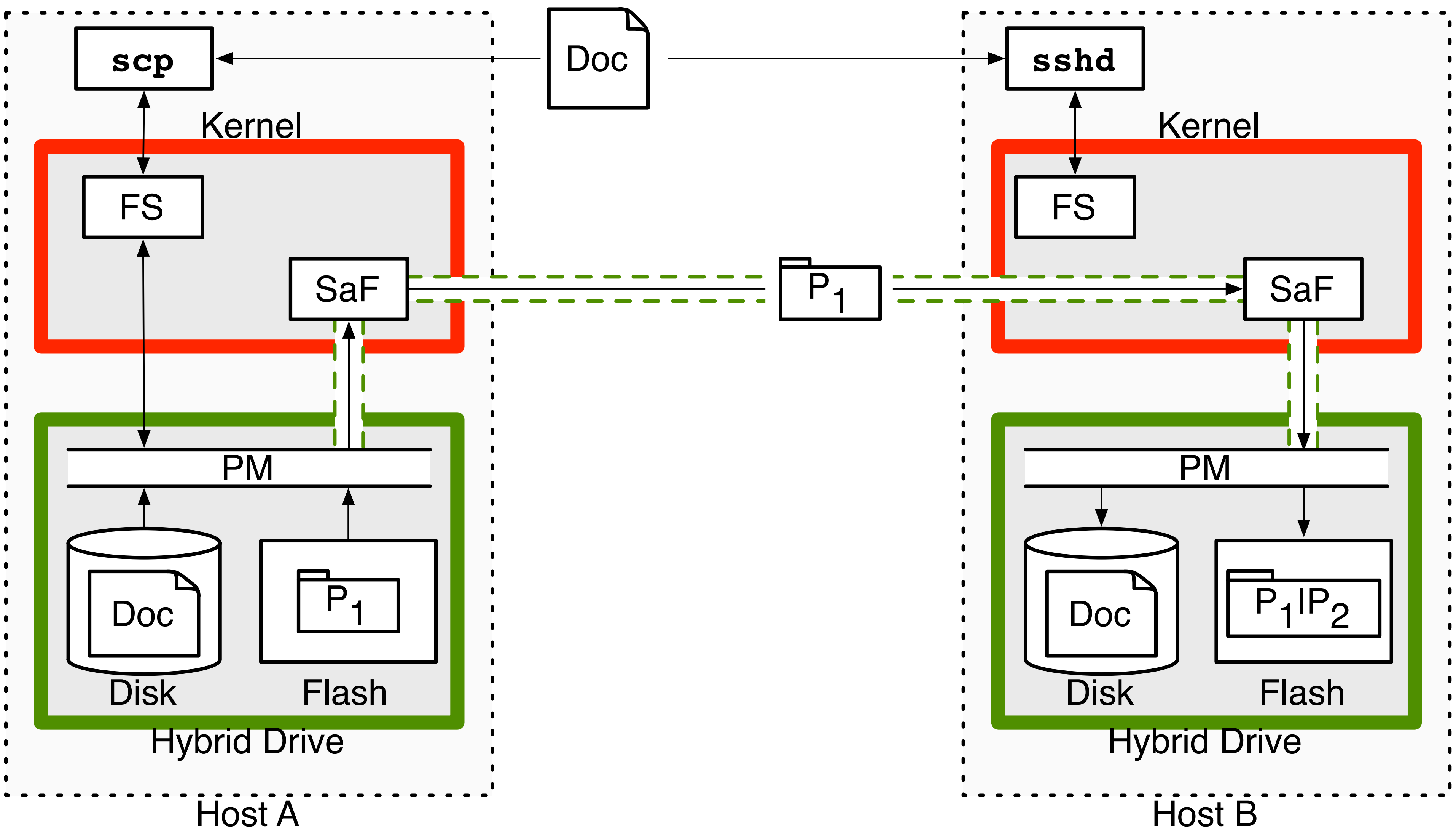
A secure tunnel is established between disks through the untrusted OS.

Distributed Example



The document is transferred as normal.

Distributed Example



The destination disk checks the integrity once the write-through is completed and appends a new provenance entry.

Distributed Provenance Overheads

PENNSTATE 1855

- Overhead increases monotonically as data is shared.
- Two implications:
  - Storage costs within a single domain
    - High sharing factor: redundant provenance data
    - Long per-host modification histories: higher redundancy factor
    - Even though document size may remain constant!
  - Audit costs between domains
    - As sharing of a document increases, the computational cost of sharing increases